

Short Papers

A Simple and Systematic Approach to Assigning Denavit–Hartenberg Parameters

Peter I. Corke

Abstract—This paper presents a simple and intuitive approach to determining the kinematic parameters of a serial-link robot in Denavit–Hartenberg (DH) notation. Once a manipulator’s kinematics is parameterized in this form, a large body of standard algorithms and code implementations for kinematics, dynamics, motion planning, and simulation are available. The proposed method has two parts. The first is the “walk through,” a simple procedure that creates a string of elementary translations and rotations, from the user-defined base coordinate to the end-effector. The second step is an algebraic procedure to manipulate this string into a form that can be factorized as link transforms, which can be represented in standard or modified DH notation. The method allows for an arbitrary base and end-effector coordinate system as well as an arbitrary zero joint angle pose. The algebraic procedure is amenable to computer algebra manipulation and a Java program is available as supplementary downloadable material.

Index Terms—Denavit–Hartenberg (DH), kinematics.

I. INTRODUCTION

The Denavit–Hartenberg (DH) notation for describing a serial-link mechanism geometry is a fundamental tool of the roboticist. Given such a description of a manipulator, we can make use of established algorithmic techniques to find kinematic solutions, Jacobians, dynamics, motion planning, and simulation, for example [1], [2].

Most modern industrial robots have a kinematic configuration similar to a Puma robot and the textbook DH parameters, with some adjustments for the particular robot, will suffice. However, determining the DH parameters and link coordinate frames for a completely new mechanism is harder than it should be, even for an experienced roboticist. Complications include spherical joints, base and tool transforms, and arbitrary world coordinate frames. The kinematic zero-angle configuration of the robot is often different to the joint controller’s zero-angle configuration, and requires that joint angle offsets be introduced.

The fundamentals of serial-link robot kinematics and the DH [3] notation are well covered in standard texts [4], [5]. Each link is represented by two parameters: the link length a_i , and link twist α_i , which define the relative location of the two attached joint axes in space. The link parameters for the first and last links are meaningless, and are arbitrarily chosen to be 0. Joints are also described by two parameters: the link offset d_i , which is the distance from one link to the next along the axis of the joint, and the joint angle θ_i , which is the rotation of one link with respect to the next about the joint axis.

Manuscript received February 18, 2007. This paper was recommended for publication by Editor F. Park upon evaluation of the reviewers’ comments.

The author is with the CSIRO ICT Centre, Brisbane, QLD 4000, Australia (e-mail: Peter.Corke@csiro.au).

This paper has supplementary multimedia material available at <http://ieeexplore.ieee.org>, provided by the author. This material includes a supplementary Java program and the robot examples included in this paper.

Digital Object Identifier 10.1109/TRO.2007.896765

For a revolute axis θ_i is the joint variable and d_i is constant, while for a prismatic joint d_i is variable, and θ_i is constant. In many of the formulations that follow we use generalized coordinates, q_i , where

$$q_i = \begin{cases} \theta_i, & \text{for a revolute joint} \\ d_i, & \text{for a prismatic joint.} \end{cases}$$

The DH representation results in a link transform matrix that transforms link coordinate frame $i - 1$ to frame i of the form

$${}^{i-1}A_i(\theta_i, d_i, a_i, \alpha_i) = R_z(\theta_i)T_z(d_i)T_x(a_i)R_x(\alpha_i) \quad (1)$$

where R_k denotes rotation about axis k and T_k denotes translation along axis k . The notation is elaborated in Section III.

For an n -link manipulator, we can express the overall robot transform in terms of the individual link transforms

$${}^0T_n = {}^0A_1{}^1A_2 \cdots {}^{n-1}A_n \quad (2)$$

which results in a string of elementary transformations

$$R_z(\theta_1)T_z(d_1)T_x(a_1)R_x(\alpha_1)R_z(\theta_2)T_z(d_2)T_x(a_2)R_x(\alpha_2) \cdots R_z(\theta_n)T_z(d_n)T_x(a_n)R_x(\alpha_n). \quad (3)$$

The classical method of assigning DH parameters is to systematically assign a coordinate frame to each link, but there are significant constraints on each frame, and also the coordinate system of the base and the end-effector. An interesting complication with the DH notation is the zero-angle configuration, that is, the pose when all joint angles are zero. For the Puma robot, this is a nonobvious “L-shaped” pose with the upper arm horizontal and the lower arm vertically upward. A robot control designer may choose the zero-angle configuration to be something more obvious, such as that shown in Fig. 2. The kinematic zero-angle configuration of the robot is often different from the joint controller’s zero-angle configuration, and requires that joint angle offsets be determined.

This paper presents a new method of generating DH parameters. It is a two-step method. The first step involves a very simple and intuitive approach to describing the manipulator kinematics as a series of elementary translations and rotations. Unlike the conventional approach, there are no constraints on the axes about which these rotations or translations can occur. The second step is the application of a set of algebraic rules that can be applied to this sequence to convert it into the DH form. Joint angle offsets, if required by the chosen zero-angle configuration chosen, are generated automatically, as are base and tool transformations.

In 1986, Craig [6] introduced the modified DH notation, where the link coordinate frame is attached to the proximal, rather than distal, end of each link, see [7] for a comparison of the two approaches. According to Craig [6]

$${}^{i-1}\bar{A}_i = R_x(\alpha_{i-1})T_x(a_{i-1})R_z(\theta_i)T_z(d_i) \quad (4)$$

which he denotes as ${}^{i-1}\bar{A}$ and has the same terms as (2) but in a different order. It is important to note that the algorithmic implementation for kinematics, Jacobians, and dynamics depends on the convention

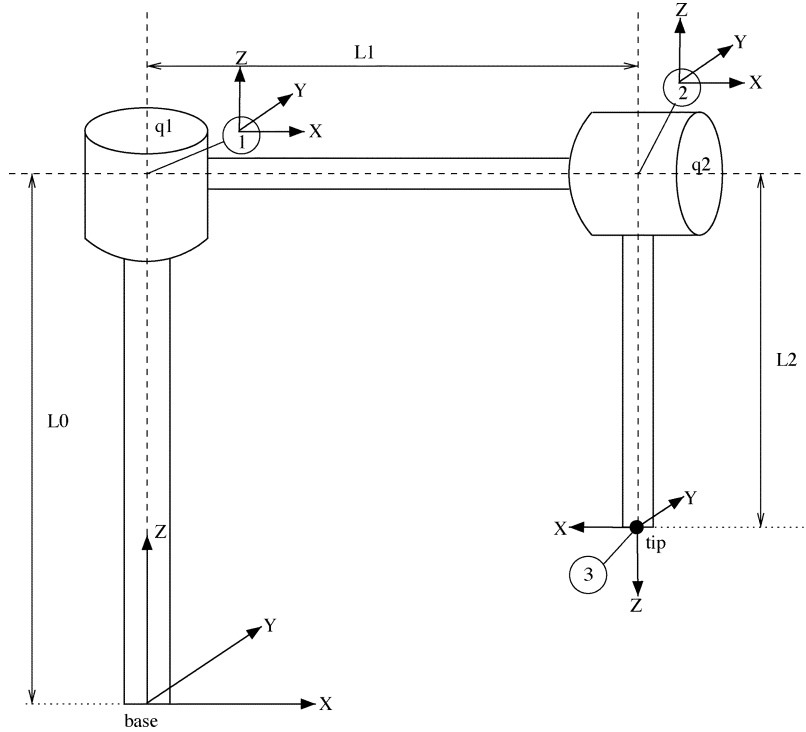


Fig. 1. Two-axis Furata pendulum example shown in its zero-angle pose.

used. The method proposed in this paper is also able to generate modified DH parameters.

The remainder of the paper is organized as follows. Section II describes the first stage, or walk-through process, for two common mechanisms: the two-axis Furata pendulum [8] and the well-known Puma 560 robot [5]. Section III describes, for these two mechanisms, how to manipulate the transform strings to generate DH parameters. An algorithm for automatically manipulating these strings is then presented. Finally, Section IV presents conclusions.

II. STEP 1: THE MANIPULATOR WALK-THROUGH

We will first introduce the notation to describe the elementary translations and rotations with which we will describe the manipulator. A pure translation of r along the current X , Y , or Z axis is

$$T_i(r), i \in \{x, y, z\}.$$

A pure rotation of θ about the current X , Y , or Z axis is

$$R_i(\theta), i \in \{x, y, z\}.$$

We will also use the shorthand notation

$$R_i \equiv R_i\left(\frac{\pi}{2}\right), i \in \{x, y, z\}$$

to represent rotations of $\pi/2$ about a particular axis. It is not important at this stage how T_i and R_i are represented, but if it assists in understanding, they could be considered as 4×4 homogeneous transformation matrices [5].

With these few preliminaries out of the way, we can tackle our first example. Consider the simple mechanism given in Fig. 1, which is known as a Furata pendulum [8]. It comprises a 1-degree-of-freedom (DOF) pendulum hanging from the end of a rotating arm. While per-

haps not typically considered as a robot, if we can describe it using DH notation, then we can use many existing tools [1], [9], [10] to generate its forward and inverse kinematic solutions, as well as its dynamic equations of motion. Further, we require that the pose shown is the zero-angle pose of the mechanism.

Let us imagine a standard right-handed coordinate frame at the base of the mechanism. Next we imagine this coordinate frame moving through the mechanism by a sequence of elementary rotations and translations. We move the frame up to ①, by translating L_0 along Z , rotating by q_1 about Z , then translating across to ② by L_1 along X , rotating by q_2 about X , then translating along Z by $-L_2$ to reach ③, where we rotate by π about the Y axis so as to have the Z axis pointing outward. It is almost trivially easy to write down

$$T_z(L_0)R_z(q_1)T_x(L_1)R_x(q_2)T_z(-L_2)R_y(\pi). \quad (5)$$

For the more complex 6-DOF Puma robot of Fig. 2, we follow a similar process. We move the base frame up to ①, translating by L_1 along Z , rotate by q_1 about Z , then translate by L_2 along Y to reach ②. We rotate by q_2 about Y , then translate by L_3 along Z to reach ③. A small translation L_6 along X , then a rotation of q_3 about X , and a translation of L_4 along X ($L_4 < 0$) brings us to ④, and so on. As we go, we have written down

$$T_z(L_1)R_z(q_1)R_y(q_2)T_y(L_2)T_z(L_3)R_y(q_3)T_x(L_6)T_y(L_4) \cdots \\ T_z(L_5)R_z(q_4)R_y(q_5)R_z(q_6)T_z(L_7) \quad (6)$$

where, again, the joint angles q_i are zero in the pose shown.

III. STEP 2: THE ALGEBRA

In this section, we will define some algebraic rules that allow us to systematically transform the strings of elementary rotations and translations into DH notation. First, we need to define some additional no-

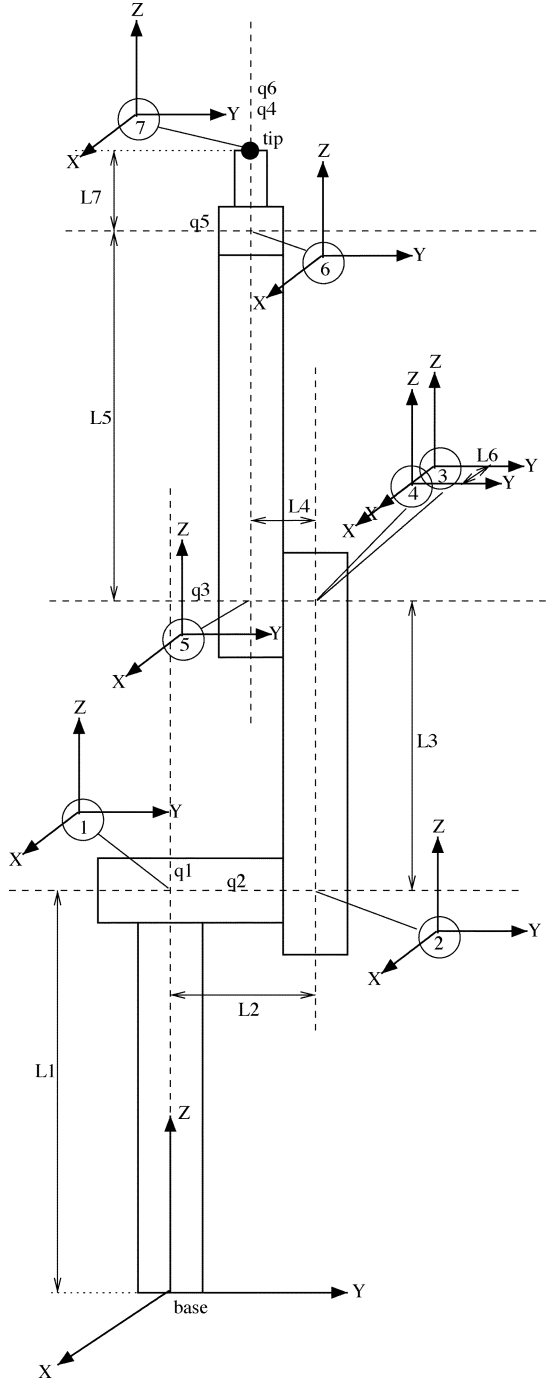


Fig. 2. Puma robot example. Side elevation showing critical dimensions. Note that L6 is a translation in the x direction between the q_2 and q_3 axes. The robot is shown in its zero-angle pose.

tation and then some transformation rules. It is useful to define the relationships

$$T'_i(r) \equiv T_i(-r) \quad (7)$$

$$R'_i(\theta) \equiv R_i(-\theta). \quad (8)$$

Commutivity applies to translation

$$T_i(r_1)T_j(r_2) \equiv T_j(r_2)T_i(r_1), i, j \in \{x, y, z\} \quad (9)$$

but not rotation

$$R_i(\theta_1)R_j(\theta_2) \neq R_j(\theta_2)R_i(\theta_1), i, j \in \{x, y, z\}, i \neq j. \quad (10)$$

Rotations obey the cyclic rules

$$R_x R_y(\theta) R'_x \equiv R_z(\theta) \quad (11)$$

$$R_y R_z(\theta) R'_y \equiv R_x(\theta) \quad (12)$$

$$R_z R_x(\theta) R'_z \equiv R_y(\theta) \quad (13)$$

and anticyclic rotation rules

$$R'_y R_x(\theta) R_y \equiv R_z(\theta) \quad (14)$$

$$R'_z R_y(\theta) R_z \equiv R_x(\theta). \quad (15)$$

Similar rotations and translations can be compounded

$$R_i(\theta_1)R_i(\theta_2) \equiv R_i(\theta_1 + \theta_2), i, j \in \{x, y, z\} \quad (16)$$

$$T_i(r_1)T_i(r_2) \equiv T_i(r_1 + r_2), i, j \in \{x, y, z\}. \quad (17)$$

For mixed rotations and translation, we can write

$$R_i(\theta)T_i(r) \equiv T_i(r)R_i(\theta), i \in \{x, y, z\} \quad (18)$$

$$R_i(\theta)T_j(r) \equiv s(k)T_k(r)R_i, i, j \in \{x, y, z\}, i \neq j \quad (19)$$

where $k = [x, y, z] \setminus \{i, j\}$, $[]$ represents an ordered set and the backslash represents the set difference operator, and

$$s(k) = \begin{cases} +1, & i > j \\ -1, & i < j \end{cases}$$

such that $y > x$ and $x > z$. For terms that involve a joint variable, we can use

$$R_x(q) \equiv R_y R_z(q) R'_y \quad (20)$$

$$R_y(q) \equiv R'_x R_z(q) R_x \quad (21)$$

$$T_x(q) \equiv R_y T_z(q) R'_y \quad (22)$$

$$T_y(q) \equiv R'_x T_z(q) R_x. \quad (23)$$

The following substitutions for fixed terms:

$$R_y \equiv R_z R_x R'_z \equiv R'_x R_z R_x \quad (24)$$

$$T_y \equiv R_z T_x R'_z \equiv R'_x T_z R_x \quad (25)$$

will be needed later. No proofs are offered for these rules, but they can be readily shown by considering R_i and T_i as a 4×4 homogeneous transformations.

A. Transformation Rules

We return now to the strings of elementary rotations and translations from Section II. Consider first the Furuta pendulum given in (5). Our first step is to push constant (not joint variable) transformations as far to the right as we can, using pairwise commutative swaps

$$\boxed{T_z(L_0)} R_z(q_1) T_x(L_1) R_x(q_2) T_z(L_2) R_x R_y(\pi) \quad (26)$$

$$R_z(q_1) \boxed{T_x(L_1)} \boxed{T_z(L_0)} R_x(q_2) T_z(L_2) R_x R_y(\pi) \quad (27)$$

$$R_z(q_1) T_z(L_0) R_x(q_2) T_z(L_2) \boxed{T_x(L_1)} R_x R_y(\pi) \quad (28)$$

$$R_z(q_1) T_z(L_0) R_x(q_2) T_z(L_2) T_x(L_1) R_x R_y(\pi) \quad (29)$$

The double box represents an initial position or value for a term, and the single box in the following line represents its final position or value.

Here $T_z(L_0)$ has crossed $R_z(q_1)$, but cannot cross $R_x(q_2)$. Similarly, $T_x(L_1)$ has crossed $R_x(q_2)$.

The DH conventions require that the axis of a joint is about or along the Z axis of the coordinate frame. To achieve this, we must substitute for the term $R_x(q_2)$ using (20)

$$R_z(q_1)T_z(L_0)\boxed{R_x(q_2)}T_z(L_2)T_x(L_1)R_xR_y(\pi) \quad (30)$$

$$R_z(q_1)T_z(L_0)\boxed{R_yR_z(q_2)R'_y}T_z(L_2)T_x(L_1)R_xR_y(\pi) \quad (31)$$

$$R_z(q_1)T_z(L_0)R_yR_z(q_2)R'_yT_z(L_2)T_x(L_1)R_xR_y(\pi) \quad (32)$$

which is starting to show some of the structure that we want, but we have introduced two undesirable R_y terms. The vertical strut is a notational convenience that partitions terms into groups associated with links. Dealing with the first link, which is the first three terms of (32), we have an R_y term which is not allowed, so we will substitute it using (21)

$$R_z(q_1)T_z(L_0)\boxed{R_y}\boxed{R_z(q_2)R'_yT_z(L_2)T_x(L_1)R_xR_y(\pi)} \quad (33)$$

$$R_z(q_1)T_z(L_0)\boxed{R_zR_xR'_z}R_z(q_2)R'_yT_z(L_2)T_x(L_1)R_xR_y(\pi) \quad (34)$$

$$R_z(q_1)T_z(L_0)\boxed{R_z}R_xR'_zR_z(q_2)R'_yT_z(L_2)T_x(L_1)R_xR_y(\pi) \quad (35)$$

$$R_z(q_1)T_z(L_0)\boxed{R_z}R_xR'_zR_z(q_2)R'_yT_z(L_2)T_x(L_1)R_xR_y(\pi) \quad (36)$$

$$\boxed{R_z(q_1)R_z}T_z(L_0)R_x\boxed{R'_zR_z(q_2)}\times R'_yT_z(L_2)T_x(L_1)R_xR_y(\pi) \quad (37)$$

$$\boxed{R_z(q_1 + \frac{\pi}{2})}T_z(L_0)R_x\boxed{R_z(q_2 - \frac{\pi}{2})}R'_yT_z(L_2)T_x(L_1)R_xR_y(\pi). \quad (38)$$

Gathering adjacent R_z terms in (38), we have automatically introduced joint angle offsets, that is, the joint variables have an associated offset that is required in order for the mechanism to have the zero-angle pose shown in Fig. 1. The first link, that is, the first three terms of (38), now has exactly the form of (1) and can be written as a DH link transform as per (1)

$$\boxed{R_z(q_1 + \frac{\pi}{2})T_z(L_0)R_x}R_z(q_2 - \frac{\pi}{2})\times R'_yT_z(L_2)T_x(L_1)R_xR_y(\pi) \quad (39)$$

$$\boxed{A(q_1 + \frac{\pi}{2}, L_0, 0, 0)}R_z(q_2 - \frac{\pi}{2})\times R'_yT_z(L_2)T_x(L_1)R_xR_y(\pi). \quad (40)$$

The remaining terms must form the second link and possibly a trailing tool transform, but it contains a nonallowed R'_y term, which we will first push as far to the right of the expression as we can

$$A(q_1 + \frac{\pi}{2}, L_0, 0, 0)R_z(q_2 - \frac{\pi}{2})\times \boxed{R'_yT_z(L_2)}T_x(L_1)R_xR_y(\pi) \quad (41)$$

$$A(q_1 + \frac{\pi}{2}, L_0, 0, 0)R_z(q_2 - \frac{\pi}{2})\times \boxed{T_x(-L_2)R'_y}T_x(L_1)R_xR_y(\pi) \quad (42)$$

$$A(q_1 + \frac{\pi}{2}, L_0, 0, 0)R_z(q_2 - \frac{\pi}{2})T_x(-L_2)\times \boxed{R'_yT_x(L_1)}R_xR_y(\pi) \quad (43)$$

$$A(q_1 + \frac{\pi}{2}, L_0, 0, 0)R_z(q_2 - \frac{\pi}{2})\times T_x(-L_2)\boxed{T_z(L_1)R'_y}R_xR_y(\pi). \quad (44)$$

As it moved to the right, it has flipped the orientation of the translation terms and also swapped their order, but this is not a

problem since they are commutative. Now we substitute $R'_y = R'_xR'_zR_x$ using (24)

$$A(q_1 + \frac{\pi}{2}, L_0, 0, 0)R_z(q_2 - \frac{\pi}{2})\times T_x(-L_2)T_z(L_1)\boxed{R'_y}R_x \quad (45)$$

$$A(q_1 + \frac{\pi}{2}, L_0, 0, 0)R_z(q_2 - \frac{\pi}{2})\times T_x(-L_2)T_z(L_1)\boxed{R'_xR'_zR_x}R_x \quad (46)$$

$$A(q_1 + \frac{\pi}{2}, L_0, 0, 0)R_z(q_2 - \frac{\pi}{2})\times T_x(-L_2)T_z(L_1)R'_xR'_zR_xR_x \quad (47)$$

$$A(q_1 + \frac{\pi}{2}, L_0, 0, 0)R_z(q_2 - \frac{\pi}{2})\times T_x(-L_2)T_z(L_1)R'_xR'_zR_x(\pi) \quad (48)$$

which is now in the desired form

$$A(q_1 + \frac{\pi}{2}, L_0, 0, 0)\boxed{R_z(q_2 - \frac{\pi}{2})T_x(-L_2)T_z(L_1)R'_x}\times R'_zR_x(\pi) \quad (49)$$

$$A(q_1 + \frac{\pi}{2}, L_0, 0, 0)A(q_2 - \frac{\pi}{2}, L_1, -L_2, -\frac{\pi}{2})\times R'_zR_x(\pi) \quad (50)$$

$$A(\tilde{q}, L_0, 0, 0)A(\tilde{q}_2, L_1, -L_2, -\frac{\pi}{2})R'_zR_x(\pi) \quad (51)$$

where \tilde{q} are the joint angles in the DH model and the offsets give the zero-angle pose, as shown in the original diagram

$$\tilde{q}_1 = q_1 + \frac{\pi}{2} \quad (52)$$

$$\tilde{q}_2 = q_2 - \frac{\pi}{2}. \quad (53)$$

A tool transform $R'_zR_x(\pi)$ has also been isolated at the right-hand side of the equation, the transforms that did not factor into DH link terms.

In (34) and (46), we have made different substitutions for R_y . This requires some choice and judgment, but a heuristic has been developed.

The Puma robot example is necessarily more complex and will be given in a more compact form in Table I. At line 11, we push the constant R_x to the right since the terms are out of order. This results in the creation of a T_y term at line 12 which is not allowed, and this is then substituted in line 15, after which, terms cancel or combine. We can see that four joints are factorized after just 11 algebraic steps.

B. An Algorithm for Automatic Symbolic Manipulation

The approach just given can be mechanized by the following rules.

- 1) Move all constant (nonjoint variable) terms as far to the right as they can go without changing any term that they cross. For instance, a translation term T_i can cross T_j , $j \in \{x, y, z\}$ or R_i . A rotation term R_i can cross any T_i . If a term meets a term of the same type, they should be merged.
- 2) For each term containing a generalized coordinate, make the following substitutions:

$$R_x(q) := R_yR_z(q)R'_y \quad (54)$$

$$R_y(q) := R_xR_z(q)R'_x \quad (55)$$

$$T_x(q) := R_yT_z(q)R'_y \quad (56)$$

$$T_y(q) := R'_xT_z(q)R_x. \quad (57)$$

- 3) Combine adjacent rotations or translation about the same axis.
- 4) Combine groups of elementary operations into A matrices for standard or modified DH notation which are, respectively

$$R_z(\theta)T_x(a)T_z(d)R_x(\alpha) := A(\theta, d, a, \alpha) \quad (58)$$

$$R_x(\alpha)T_x(a)R_z(\theta)T_z(d) := \tilde{A}(\theta, d, a, \alpha). \quad (59)$$

Swap terms using commutation laws as appropriate.

- 5) Push R_y terms as far to the right within the link group as possible, that is, no further than the next joint variable term. As the term

TABLE I
SYMBOLIC MANIPULATION FOR 6-DOF PUMA EXAMPLE

1	$T_z(L_1) R_z(q_1) R_y(q_2) T_y(L_2) T_z(L_3) R_y(q_3) T_x(L_6) T_y(L_4) T_z(L_5) R_z(q_4) R_y(q_5) R_z(q_6)$	
2	$R_z(q_1) T_z(L_1) R_y(q_2) T_y(L_2) T_z(L_3) R_y(q_3) T_x(L_6) T_y(L_4) T_z(L_5) R_z(q_4) R_y(q_5) R_z(q_6)$	push $T_z(L_1)$ to right
3	$R_z(q_1) T_z(L_1) R_y(q_2) T_z(L_3) R_y(q_3) T_x(L_6) T_y(L_2) T_y(L_4) T_z(L_5) R_z(q_4) R_y(q_5) R_z(q_6)$	push $T_y(L_2)$ to right
4	$R_z(q_1) T_z(L_1) R_y(q_2) T_z(L_3) R_y(q_3) T_x(L_6) T_y(L_2) T_y(L_4) T_z(L_5) R_z(q_4) R_y(q_5) R_z(q_6)$	push $T_y(L_2)$ to right
5	$R_z(q_1) T_z(L_1) R_y(q_2) T_z(L_3) R_y(q_3) T_x(L_6) T_y(L_2 + L_4) T_z(L_5) R_z(q_4) R_y(q_5) R_z(q_6)$	merge L_2 and L_4
6	$R_z(q_1) T_z(L_1) R_y(q_2) T_z(L_3) R_y(q_3) T_y(L_2 + L_4) T_z(L_5) T_x(L_6) R_z(q_4) R_y(q_5) R_z(q_6)$	push $T_x(L_6)$ to right
7	$R_z(q_1) T_z(L_1) R_y(q_2) T_z(L_3) R_y(q_3) T_y(L_2 + L_4) T_x(L_6) R_z(q_4) T_z(L_5) R_y(q_5) R_z(q_6)$	push $T_z(L_5)$ to right
8	$R_z(q_1) T_z(L_1) R_y(q_2) T_z(L_3) R_y(q_3) T_y(L_2 + L_4) T_x(L_6) R_z(q_4) T_z(L_5) R_y(q_5) R_z(q_6)$	push $T_z(L_5)$ to right
9	$R_z(q_1) T_z(L_1) R'_x R_z(q_2) R_x T_z(L_3) R'_x R_z(q_3) R_x T_y(L_2 + L_4) T_x(L_6) R_z(q_4) T_z(L_5) R'_x R_z(q_5) R_x R_z(q_6)$	substitute $R_y(q)$
10	$R_z(q_1) T_z(L_1) R'_x R_z(q_2) R_x T_z(L_3) R'_x R_z(q_3) R_x T_y(L_2 + L_4) T_x(L_6) R_z(q_4) T_z(L_5) R'_x R_z(q_5) R_x R_z(q_6)$	
11	$A(q_1, L_1, 0, \frac{\pi}{2}) R_z(q_2) R_x T_z(L_3) R'_x R_z(q_3) R_x T_y(L_2 + L_4) T_x(L_6) A(q_4, L_5, 0, -\frac{\pi}{2}) A(q_5, 0, 0, \frac{\pi}{2}) A(q_6, 0, 0, 0)$	factorize
12	$A(q_1, L_1, 0, \frac{\pi}{2}) R_z(q_2) T_y(-L_3) R_x R'_x R_z(q_3) R_x T_y(L_2 + L_4) T_x(L_6) A(q_4, L_5, 0, -\frac{\pi}{2}) A(q_5, 0, 0, \frac{\pi}{2}) A(q_6, 0, 0, 0)$	push R_x right
13	$A(q_1, L_1, 0, \frac{\pi}{2}) R_z(q_2) T_y(-L_3) R_x R'_x R_z(q_3) R_x T_y(L_2 + L_4) T_x(L_6) A(q_4, L_5, 0, -\frac{\pi}{2}) A(q_5, 0, 0, \frac{\pi}{2}) A(q_6, 0, 0, 0)$	
14	$A(q_1, L_1, 0, \frac{\pi}{2}) R_z(q_2) T_y(-L_3) I R_z(q_3) R_x T_y(L_2 + L_4) T_x(L_6) A(q_4, L_5, 0, -\frac{\pi}{2}) A(q_5, 0, 0, \frac{\pi}{2}) A(q_6, 0, 0, 0)$	cancel
15	$A(q_1, L_1, 0, \frac{\pi}{2}) R_z(q_2) R_z T_x(-L_3) R'_z R_z(q_3) R_x T_y(L_2 + L_4) T_x(L_6) A(q_4, L_5, 0, -\frac{\pi}{2}) A(q_5, 0, 0, \frac{\pi}{2}) A(q_6, 0, 0, 0)$	subst T_y
16	$A(q_1, L_1, 0, \frac{\pi}{2}) R_z(q_2) R_z T_x(-L_3) R'_z R_z(q_3) R_x T_y(L_2 + L_4) T_x(L_6) A(q_4, L_5, 0, -\frac{\pi}{2}) A(q_5, 0, 0, \frac{\pi}{2}) A(q_6, 0, 0, 0)$	
17	$A(q_1, L_1, 0, \frac{\pi}{2}) R_z(q_2 + \frac{\pi}{2}) T_x(-L_3) R_z(q_3 - \frac{\pi}{2}) R_x T_y(L_2 + L_4) T_x(L_6) A(q_4, L_5, 0, -\frac{\pi}{2}) A(q_5, 0, 0, \frac{\pi}{2}) A(q_6, 0, 0, 0)$	merge R_z
18	$A(q_1, L_1, 0, \frac{\pi}{2}) R_z(q_2 + \frac{\pi}{2}) T_x(-L_3) R_z(q_3 - \frac{\pi}{2}) R_x T_y(L_2 + L_4) T_x(L_6) A(q_4, L_5, 0, -\frac{\pi}{2}) A(q_5, 0, 0, \frac{\pi}{2}) A(q_6, 0, 0, 0)$	merge R_z
19	$A(q_1, L_1, 0, \frac{\pi}{2}) A(q_2, 0, -L_3, \frac{\pi}{2}) R_z(q_3 - \frac{\pi}{2}) R_x T_y(L_2 + L_4) T_x(L_6) A(q_4, L_5, 0, -\frac{\pi}{2}) A(q_5, 0, 0, \frac{\pi}{2}) A(q_6, 0, 0, 0)$	factorize
20	$A(q_1, L_1, 0, \frac{\pi}{2}) A(q_2, 0, -L_3, \frac{\pi}{2}) R_z(q_3 - \frac{\pi}{2}) T_z(L_2 + L_4) R_x T_x(L_6) A(q_4, L_5, 0, -\frac{\pi}{2}) A(q_5, 0, 0, \frac{\pi}{2}) A(q_6, 0, 0, 0)$	push R_x right
21	$A(q_1, L_1, 0, \frac{\pi}{2}) A(q_2, 0, -L_3, \frac{\pi}{2}) R_z(q_3 - \frac{\pi}{2}) T_z(L_2 + L_4) T_x(L_6) R_x A(q_4, L_5, 0, -\frac{\pi}{2}) A(q_5, 0, 0, \frac{\pi}{2}) A(q_6, 0, 0, 0)$	push R_x right
22	$A(q_1, L_1, 0, \frac{\pi}{2}) A(q_2, 0, -L_3, \frac{\pi}{2}) R_z(q_3 - \frac{\pi}{2}) T_z(L_2 + L_4) T_x(L_6) R_x A(q_4, L_5, 0, -\frac{\pi}{2}) A(q_5, 0, 0, \frac{\pi}{2}) A(q_6, 0, 0, 0)$	
23	$A(q_1, L_1, 0, \frac{\pi}{2}) A(q_2, 0, -L_3, \frac{\pi}{2}) A(q_3 - \frac{\pi}{2}, L_2 + L_4, L_6, \frac{\pi}{2}) A(q_4, L_5, 0, -\frac{\pi}{2}) A(q_5, 0, 0, \frac{\pi}{2}) A(q_6, 0, 0, 0)$	factorize

moves rightward, it will flip the translation direction of T_i terms it crosses, according to (19).

6) Substitute

$$R_y := R_z R_x R'_z \quad (60)$$

$$R_y := R'_x R_z R_x \quad (61)$$

$$T_y := R_z T_x R'_z \quad (62)$$

$$T_y := R'_x T_z R_x. \quad (63)$$

The appropriate substitution to employ depends on the adjacent terms, with a stronger preference to conform to an adjacent joint variable term. For example, $R_z(q) R_y$ should use the substitution (60).

7) Repeat steps 3–6 until no more transformations occur.

Such rules can be very easily coded in a symbolic processing computer language such as Python, Tcl, Maple, or Lisp. We have included a supplementary Java program and the robot examples used in this paper. This is available at <http://www.ieeexplore.ieee.org>.

IV. CONCLUSION

This paper has presented a simple approach to determining the kinematic parameters of a serial-link mechanism in either standard or modified DH notation. The method has two parts. The first is the walk-through, a simple procedure that creates a string of elementary translations and rotations from the user-defined base coordinates to

the end-effector. The second step is an algebraic procedure to manipulate this string into a form that can be factorized as link transforms which can be represented in standard or modified DH notation, and automatically provides the kinematic joint angle offsets. The algebraic procedure is amenable to computer algebra manipulation using languages with list processing capabilities such as Python, Tcl, Maple, or Lisp.

REFERENCES

- [1] J. Nethery and M. W. Spong, "Robotica: A Mathematica package for robot analysis," *IEEE Robot. Autom. Mag.*, vol. 1, pp. 13–20, Mar. 1994.
- [2] P. Corke, "A robotics toolbox for MATLAB," *IEEE Robot. Autom. Mag.*, vol. 3, pp. 24–32, Sep. 1996.
- [3] R. S. Hartenberg and J. Denavit, "A kinematic notation for lower pair mechanisms based on matrices," *J. Appl. Mech.*, vol. 77, no. 2, pp. 215–221, Jun. 1955.
- [4] M. Spong, S. Hutchinson, and M. Vidyasagar, *Robot Modeling and Control*. New York: Wiley, 2006.
- [5] R. P. Paul, *Robot Manipulators: Mathematics, Programming, and Control*. Cambridge, MA: MIT Press, 1981.
- [6] J. J. Craig, *Introduction to Robotics*. Reading, MA: Addison Wesley, 1986.
- [7] H. Lipkin, "A note on Denavit–Hartenberg notation in robotics," in *Proc. ASME IDETC/CIE*, Long Beach, CA, Sep. 2005, pp. 921–926.
- [8] K. Astrom and K. Furuta, "Swinging-up a pendulum," *Automatica*, vol. 36, pp. 287–295, 2000.
- [9] "Matlab User's Guide," The MathWorks, Inc., Natick, MA, Jan. 1990.
- [10] B. Char et al., *Maple V Language Reference Manual*. New York: Springer-Verlag, 1991.